

PROJECT DEVELOPMENT PHASE

SPRINT4

Date	24 October 2022
Team ID	PNT2022TMID17788
Project Name	Smart Lender - Applicant Credibility Prediction for Loan Approval
Maximum Marks	20 Marks

APPLICATION DEVELOPMENT

IBM Watson Studio

Search in your workspaces

Buy

Anandha Kannan N's Acco...

Dallas

40%

Projects / Smart Lender / Smart Lender

File Edit View Insert Cell Kernel Help

Not Trusted | Python 3.9

In [252]: #Importing The Libraries

In [253]:

```
import pandas as pd
import numpy as np
import pickle
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, f1_score
from sklearn.linear_model import LogisticRegression
import os, types
import pandas as pd
from boto3.client import Config
import boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = boto3.client(service_name='s3',
    aws_access_key_id='9MSu1qb23M43240Dhw50tefIvzxX_1pKtG0mXGKypw',
    aws_secret_access_key='ibm_auth_endpoint=https://iam.cloud.ibm.com/oidc/token',
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'smartlender-donotdelete-pr-cmau87ubqljq'
object_key = 'train_us10uk_Ctu291.csv'

body = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, '__iter__'): body.__iter__ = types.MethodType(__iter__, body)

data = pd.read_csv(body)
```

In [254]: data

Out[254]:

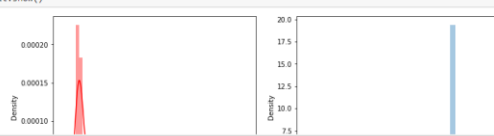
	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1506.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y
...
609	LP002978	Female	No	0	Graduate	No	2900	0.0	71.0	360.0	1.0	Rural	Y
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0	40.0	180.0	1.0	Rural	Y
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0	253.0	360.0	1.0	Urban	Y
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0	187.0	360.0	1.0	Urban	Y
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0	133.0	360.0	0.0	Semurban	N

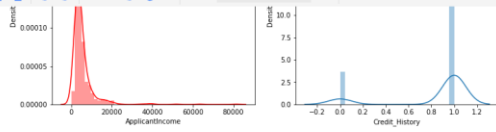
614 rows x 13 columns

In [255]: #Univariate Analysis

In [256]:

```
plt.figure(figsize=(12,5))
plt.subplot(121)
sns.distplot(data['ApplicantIncome'], color='r')
plt.subplot(122)
sns.distplot(data['Credit_History'])
plt.show()
```

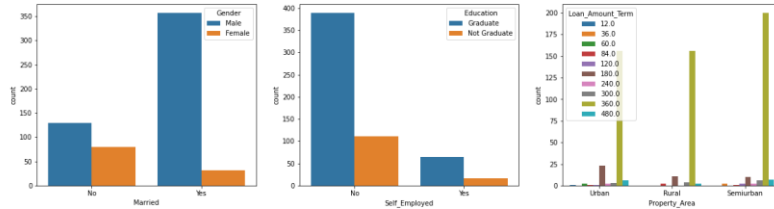




In [257]: #Bivariate Analysis

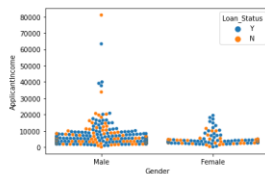
```
In [258]: plt.figure(figsize=(20,5))
plt.subplot(131)
sns.countplot(data['Married'], hue=data['Gender'])
plt.subplot(132)
sns.countplot(data['Self_Employed'], hue=data['Education'])
plt.subplot(133)
sns.countplot(data['Property_Area'], hue=data['Loan_Amount_Term'])
```

Out[258]: <AxesSubplot: xlabel='Property_Area', ylabel='count'>



In [259]: #Multivariate Analysis

```
In [260]: sns.swarmplot(data['Gender'], data['ApplicantIncome'], hue = data['Loan_Status'])
Out[260]: <AxesSubplot: xlabel='Gender', ylabel='ApplicantIncome'>
```



In [261]: #Descriptive Analysis

In [262]: data.describe()

```
Out[262]:
```

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	592.000000	600.000000	564.000000
mean	5403.459283	1621.245798	148.412162	342.000000	0.842199
std	6109.041673	2926.248369	85.587325	65.12041	0.364878
min	150.000000	0.000000	9.000000	12.000000	0.000000
25%	2877.500000	0.000000	100.000000	360.000000	1.000000
50%	3812.500000	1188.500000	128.000000	360.000000	1.000000
75%	5795.000000	2297.250000	188.000000	360.000000	1.000000
max	81000.000000	41667.000000	700.000000	480.000000	1.000000

In [263]: #Checking For Null Values

In [263]: #Checking For Null Values

In [264]: data.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
#   Column                Non-Null Count  Dtype
---  --
0   Loan_ID                614 non-null   object
1   Gender                 601 non-null   object
2   Married                611 non-null   object
3   Dependents             599 non-null   object
4   Education              614 non-null   object
5   Self_Employed          592 non-null   object
6   ApplicantIncome        614 non-null   int64
7   CoapplicantIncome      614 non-null   float64
8   LoanAmount             592 non-null   float64
9   Loan_Amount_Term       600 non-null   float64
10  Credit_History         564 non-null   float64
11  Property_Area          614 non-null   object
12  Loan_Status            614 non-null   object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.5+ KB
```

In [265]: #Data preprocessing

In [266]: data.drop(["Loan_ID"], axis=1, inplace=True)

In [267]: data

```
Out[267]:
```

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

IBM Watson Studio

Search in your workspaces

Buy

Anandha Kannan N's Acco...

Dallas

AK

Projects / Smart Lender / Smart Lender

File Edit View Insert Cell Kernel Help

Not Trusted | Python 3.9

In [268]: #Handling Missing Values

In [269]: def missing_values(df):
a = num_null_values = df.isnull().sum()
return a

In [270]: missing_values(data)

Out[270]:

Gender 13
Married 3
Dependents 15
Education 0
Self_Employed 32
ApplicantIncome 0
CoapplicantIncome 0
LoanAmount 22
Loan_Amount_Term 14
Credit_History 50
Property_Area 0
Loan_Status 0
dtype: int64

In [271]: #Handling Missing Values

In [272]: # dropping the missing values
data = data.dropna()

In [273]: data

Out[273]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y
5	Male	Yes	2	Graduate	Yes	5417	4196.0	267.0	360.0	1.0	Urban	Y
...
609	Female	No	0	Graduate	No	2900	0.0	71.0	360.0	1.0	Rural	Y

IBM Watson Studio

Search in your workspaces

Buy

Anandha Kannan N's Acco...

Dallas

AK

Projects / Smart Lender / Smart Lender

File Edit View Insert Cell Kernel Help

Not Trusted | Python 3.9

In [274]: data.isnull().sum()

Out[274]:

Gender 0
Married 0
Dependents 0
Education 0
Self_Employed 0
ApplicantIncome 0
CoapplicantIncome 0
LoanAmount 0
Loan_Amount_Term 0
Credit_History 0
Property_Area 0
Loan_Status 0
dtype: int64

In [275]: #Encoding

In [276]: from sklearn.preprocessing import OrdinalEncoder

ord_enc = OrdinalEncoder()
data[['Gender','Married','Dependents','Education','Self_Employed','Property_Area','Loan_Status']] = ord_enc.fit_transform(data[['Gender','Married','Dependents','Education','Self_Employed','Property_Area','Loan_Status']])
data.head()

Out[276]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	1.0	1.0	1.0	0.0	0.0	4583	1508.0	128.0	360.0	1.0	0.0	0.0
2	1.0	1.0	0.0	0.0	1.0	3000	0.0	66.0	360.0	1.0	2.0	1.0
3	1.0	1.0	0.0	1.0	0.0	2583	2358.0	120.0	360.0	1.0	2.0	1.0
4	1.0	0.0	0.0	0.0	0.0	6000	0.0	141.0	360.0	1.0	2.0	1.0
5	1.0	1.0	2.0	0.0	1.0	5417	4196.0	267.0	360.0	1.0	2.0	1.0

In [277]: data[['Gender','Married','Dependents','Education','Self_Employed','CoapplicantIncome','LoanAmount','Loan_Amount_Term','Credit_History','Property_Area','Loan_Status']] = data[['Gender','Married','Dependents','Education','Self_E

In [278]: data

Out[278]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	1	1	1	0	0	4583	1508	128	360	1	0	0

IBM Watson Studio

Search in your workspaces

Buy

Anandha Kannan N's Acco...

Dallas

AK

Projects / Smart Lender / Smart Lender

File Edit View Insert Cell Kernel Help

Not Trusted | Python 3.9

In [278]: data

Out[278]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	1	1	1	0	0	4583	1508	128	360	1	0	0
2	1	1	0	0	1	3000	0	66	360	1	2	1
3	1	1	0	1	0	2583	2358	120	360	1	2	1
4	1	0	0	0	0	6000	0	141	360	1	2	1
5	1	1	2	0	1	5417	4196	267	360	1	2	1
...
609	0	0	0	0	0	2900	0	71	360	1	0	1
610	1	1	3	0	0	4106	0	40	180	1	0	1
611	1	1	1	0	0	8072	240	253	360	1	2	1
612	1	1	2	0	0	7583	0	187	360	1	2	1
613	0	0	0	0	1	4563	0	133	360	0	1	0

480 rows x 12 columns

In [279]: #Train Test Split

In [280]: X = data.drop('Loan_Status', axis=1)
y = data['Loan_Status']

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2, random_state=2)
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)

(384, 11)
(384,)
(96, 11)
(96,)

In [281]: #Naive Bayes

In [282]: from sklearn.naive_bayes import GaussianNB

```
IBM Watson Studio Search in your workspaces Buy Anandha Kannan N's Acco... Dallas AN

Projects / Smart Lender / Smart Lender

File Edit View Insert Cell Kernel Help Not Trusted | Python 3.9

In [281]: #Naive Bayes

In [282]: from sklearn.naive_bayes import GaussianNB
gfc = GaussianNB()
gfc.fit(X_train, y_train)
pred1 = gfc.predict(X_test)
training_data_accuracy_nb = accuracy_score(pred1,y_test)

In [283]: training_data_accuracy_nb
Out[283]: 0.7708333333333334

In [284]: #SVM Classifier

In [285]: from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
classifier = SVC(kernel='linear')

In [286]: training the support Vector Machine model
classifier.fit(X_train,y_train)
X_test_prediction = classifier.predict(X_test)
test_data_accuracy_svm = accuracy_score(X_test_prediction,y_test)

In [287]: test_data_accuracy_svm

In [288]: #XGB Classifier

In [289]: from xgboost import XGBClassifier
xgb = XGBClassifier(learning_rate =0.1,
n_estimators=1000,
max_depth=3,
min_child_weight=1,
gamma=0,
subsample=0.8,
colsample_bytree=0.9,
silent=False, #Silent is True by default

In [291]: from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import RandomizedSearchCV
def randomized_search(params, runs=20, clf=DecisionTreeClassifier(random_state=2)):
    rand_clf = RandomizedSearchCV(clf, params, n_iter=runs, cv=5, n_jobs=-1, random_state=2)
    rand_clf.fit(X_train, y_train)
    best_model = rand_clf.best_estimator_
    # Extract best score
    best_score = rand_clf.best_score_
    # Print best score
    print("Training score: {:.3f}".format(best_score))
    # Predict test set labels
    y_pred = best_model.predict(X_test)
    # Compute accuracy
    accuracy = accuracy_score(y_test, y_pred)
    # Print accuracy
    print("Test score: {:.3f}".format(accuracy))
    return best_model

In [292]: ds = DecisionTreeClassifier(max_depth=8, max_features=0.9, max_leaf_nodes=30,
min_impurity_decrease=0.05, min_samples_leaf=0.02,
min_samples_split=10, min_weight_fraction_leaf=0.005,
random_state=2, splitter='random')
ds.fit(X_train, y_train)
pred4 = ds.predict(X_test)
test_data_accuracy_dt = accuracy_score(y_test, pred4)

In [312]: ds
Out[312]: DecisionTreeClassifier(max_depth=8, max_features=0.9, max_leaf_nodes=30,
min_impurity_decrease=0.05, min_samples_leaf=0.02,
min_samples_split=10, min_weight_fraction_leaf=0.005,
random_state=2, splitter='random')

In [293]: test_data_accuracy_dt

In [294]: from sklearn.ensemble import RandomForestClassifier
randomized_search(params={
    'min_samples_leaf':[1,2,4,6,8,10,20,30],
    'min_impurity_decrease':[0.0, 0.01, 0.05, 0.10, 0.15, 0.2],
    'max_features':['auto', 0.5, 0.7, 0.6, 0.5, 0.4],
    'max_depth':[None,2,4,6,8,10,20],
}, clf=RandomForestClassifier(random_state=2))
Training score: 0.815
Test score: 0.781
Out[294]: RandomForestClassifier(max_depth=2, max_features=0.5,
min_impurity_decrease=0.01, min_samples_leaf=10,
random_state=2)

In [295]: logisticRegr = LogisticRegression()
logisticRegr.fit(X_train, y_train)
predictions = logisticRegr.predict(X_test)
test_data_accuracy_lr = accuracy_score(y_test,predictions)

In [296]: test_data_accuracy_lr
Out[296]: 0.75

In [297]: #Knn

In [298]: from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=7)
knn.fit(X_train, y_train)
predictions = knn.predict(X_test)
test_data_accuracy_knn = accuracy_score(y_test,predictions)

In [299]: test_data_accuracy_knn
Out[299]: 0.6458333333333334

In [300]: data=[training_data_accuracy_nb,test_data_accuracy_svm,test_data_accuracy_xgb,test_data_accuracy_dt,test_data_accuracy_lr,test_data_accuracy_knn]
data = np.array(data)
data2 = data
print(data2)
```

```
IBM Watson Studio
Search in your workspaces
Buy
Analytics
Santosh K. N. Acco...
Dallia...
Python 3.9
Not Trusted

Projects / Smart Lender / Smart Lender

File Edit View Insert Cell Kernel Help

In [304]: !pip install ibm_watson_machine_learning

Requirement already satisfied: ibm_watson_machine_learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2022.9.24)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.1.3)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.26.7)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (4.8.2)
Requirement already satisfied: pandas in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.3.3)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.8.9)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.26.0)
Requirement already satisfied: pandas<1.5.0, >=24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.3.4)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: ibm-cos-sdk==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: jmespath<0.6.0, >=7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0, >=24.2->ibm_watson_machine_learning) (0.6.0)
Requirement already satisfied: python-dateutil<13.0.0, >=1.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (2.8.2)
Requirement already satisfied: pytz<2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0, >=24.2->ibm_watson_machine_learning) (2021.3)
Requirement already satisfied: numpy<1.17.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0, >=24.2->ibm_watson_machine_learning) (1.20.3)
Requirement already satisfied: six<1.15 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-dateutil<13.0.0, >=1.1->ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (1.15.0)
Requirement already satisfied: idna<4, >2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->ibm_watson_machine_learning) (3.3)
Requirement already satisfied: charset-normalizer<2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->ibm_watson_machine_learning) (2.0.4)
Requirement already satisfied: zip==0.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from importlib-metadata->ibm_watson_machine_learning) (3.6.0)
Requirement already satisfied: pyrsnap==3.0.5, >=2.0.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from packaging->ibm_watson_machine_learning) (3.0.4)

In [325]: from ibm_watson_machine_learning import APIClient
import json
import numpy as np
url_credentials = {
    "url": "https://us-south-ml.cloud.ibm.com",
    "apikey": "AR13V6F08R8Yqdg07G1GcfJf0AEZ_Cq34Q0W4v6"
}
ml_client = APIClient(url_credentials)
ml_client.spaces.list()

Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50

ID NAME CREATED
824927c5-d8c8-458e-a580-f3e6dbcb376 SmartLenderDeployment 2022-11-15T15:03:17.200Z
.....

In [326]: SPACE_ID = "824927c5-d8c8-458e-a580-f3e6dbcb376"
```

The screenshot displays the IBM Watson Studio interface. At the top, there's a navigation bar with 'Buy', 'Anandha Kannan N's Account', and 'Dallas'. Below this, the 'Projects / Smart Lender / Smart Lender' breadcrumb is visible. The main area shows a Jupyter Notebook with the following code cells:

```
In [327]: wml_client.set_default_space(SPACE_ID)
Out[327]: 'SUCCESS'
```

```
In [328]: MODEL_NAME = 'Smartlender'
DEPLOYMENT_NAME = 'SmartLender Model Deployment'
BEST_MODEL = model
```

```
In [329]: # Set Python Version
software_spec_uid = wml_client.software_specifications.get_id_by_name('runtime-22.1-py3.9')

# Setup model meta
model_props = {
    wml_client.repository.ModelMetaNames.NAME: MODEL_NAME,
    wml_client.repository.ModelMetaNames.TYPE: 'scikit-learn_1.0',
    wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}
```

```
In [330]: model_props
Out[330]: {'name': 'Smartlender',
           'type': 'scikit-learn_1.0',
           'software_spec': '12b83a17-24d8-5082-900f-0ab31fbfd1cb'}
```

```
In [331]: #Save model
model_details = wml_client.repository.store_model(model=BEST_MODEL,meta_props=model_props,training_data=X_train,training_target=y_train.values.ravel())
```

```
In [332]: model_details
Out[332]: {'entity': {'hybrid_pipeline_software_specs': [],
                    'label_column': 'l1',
                    'schemas': [{'input': [{'fields': [{'name': 'Gender', 'type': 'int64'},
{'name': 'Married', 'type': 'int64'},
{'name': 'Dependents', 'type': 'int64'},
{'name': 'Education', 'type': 'int64'},
{'name': 'Self Employed', 'type': 'int64'},
{'name': 'ApplicantIncome', 'type': 'int64'},
{'name': 'CoapplicantIncome', 'type': 'int64'},
{'name': 'LoanAmount', 'type': 'int64'},
{'name': 'Loan Amount Tail', 'type': 'int64'}]}]}]}
```

```
IBM Watson Studio Search in your workspaces Buy Anandha Kannan N's Acco... Dallas Ak

Projects / Smart Lender / Smart Lender
File Edit View Insert Cell Kernel Help
Not Trusted | Python 3.9

type: 'sklearn.linear_model',
'software_spec': '12b83a17-24d8-5082-900f-8ab31fbfd3cb'}

In [331]: #Save model
model_details = wml_client.repository.store_model(model=BEST_MODEL,meta_props=model_props,training_data=X_train,training_target=y_train.values.ravel())

In [332]: model_details

Out[332]: {'entity': {'hybrid_pipeline_software_specs': [],
  'label_column': 'li',
  'schemas': {'input': [{'name': 'Gender', 'type': 'int64'},
    {'name': 'Married', 'type': 'int64'},
    {'name': 'Dependents', 'type': 'int64'},
    {'name': 'Education', 'type': 'int64'},
    {'name': 'Self_Employed', 'type': 'int64'},
    {'name': 'ApplicantIncome', 'type': 'int64'},
    {'name': 'CoapplicantIncome', 'type': 'int64'},
    {'name': 'LoanAmount', 'type': 'int64'},
    {'name': 'Loan_Amount_Term', 'type': 'int64'},
    {'name': 'Credit_History', 'type': 'int64'},
    {'name': 'Property_Area', 'type': 'int64'}],
    'id': '1',
    'type': 'struct'},
  'output': []},
  'software_spec': {'id': '12b83a17-24d8-5082-900f-8ab31fbfd3cb',
    'name': 'runtime-22.1-py3.9'},
  'type': 'scikit-learn-1.0'},
  'metadata': {'created_at': '2022-11-15T18:01:32.671Z',
    'id': 'f3b55b44-c179-4490-adcb-70406c7b7047',
    'modified_at': '2022-11-15T18:01:35.984Z',
    'name': 'SmartLender',
    'owner': 'IBMid-666802FYOS',
    'resource_key': 'ed648b40-a469-4341-9298-d9c1e14869da',
    'space_id': '83497c5-08c8-458e-a580-f3e5d6bc3b76'},
  'system': {'warnings': []}}

In [333]: model_uid = wml_client.repository.get_model_uid(model_details);
model_uid

This method is deprecated, please use get_model_id()

Out[333]: 'f3b55b44-c179-4490-adcb-70406c7b7047'
```

MODEL DEPLOYMENT IN IBM WATSON

IBM Watson Studio Search in your workspaces Buy Anandha Kannan N's Acco... Dallas Ak

Deployments /

Online deployments
1 space

Deployed Asset type Space tags Spaces

Find deployments

Name	Asset type	Status	Space	Copies	Last updated	Created
Smart_Lender_Model_Deplo...	Model	Deployed	SmartLenderDeployment	1	Nov 15, 2022, 11:51 PM	Nov 15, 2022, 11:51 PM